

## A para-virtual SDN simulation system based on Emulab

Bo Luo, Qi Li, Yanhui Guo

Beijing University of Posts and Telecommunications, Beijing, China

luobo123@163.com

**Keywords:** SDN, Simulation System, Para-virtual, OpenFlow, Emulab

**Abstract.** With the arrival of big data era, traditional network is difficult to meet the increasing data processing need. In order to solve the problem, software defined network emerge. SDN is a revolutionary improvement compared to the traditional network, it can greatly meet the giant data processing need. However, It is really expensive to build SDN with hardware devices, so we need to develop SDN simulation system. However, the existing simulation systems have so many problems. In this paper, we propose a para-virtual simulation system based on Emulab which can overcome the problems in existing simulation systems. In the paper, we first study the SDN architecture and analyze problems in existing simulation systems. Next we put forward para-virtual SDN simulation system based on Emulab. Thirdly, we build para-virtual simulation system. Then, we carry out verification test on the simulation system we build. Finally, we make a comparison of para-virtual simulation system with the two existing simulation systems (Mininet and the pure physical simulation system).

### Introduction

With the arrival of the big data era, the network is to process increasing data. Unfortunately traditional network can no longer meet the large data processing demand. Therefore, it is urgent to find a more efficient network. Thus Software Defined Network (SDN) came into being. SDN[1] is a revolutionary improvement compared to the traditional network, it can greatly meet the giant data processing needs which is really promising. SDN separate network control and data forwarding functions which make network control and data forwarding behavior programmable.

Now there exist two SDN simulation systems, one is Lightweight virtual simulation system and the other is Full physical simulation system. Unfortunately, the two simulation systems have certain shortcomings, so in this paper we proposed a new simulation system with the concept of para-virtual which introduce the Emulab[2] technology. It can solve the performance issue caused by Lightweight virtual simulation system, while get over the management problem caused by Full physical simulation system.

### SDN / OpenFlow architecture

**SDN architecture.** The lower two layers are responsible for data forwarding while the upper two layers are responsible for controlling network traffic via OpenFlow protocol. This separate the network control rights and data forwarding rights.

**OpenFlow protocol.** Hierarchical decoupling architecture uses OpenFlow protocol[3] to separate network control and forwarding layer. OpenFlow protocol is used to regulate the communication flow between the controller and the switch, it also manage the interfaces used.

### SDN simulation systems

**Mininet virtual simulation system.** Mininet[4] simulation system builds the virtual architecture based on physical hardware in the host. This allows virtual computer reaches a maximum performance. However, Mininet performs poor when using in professional research or in commercial. It is low efficient, has less flexibility and importantly it has no graphical user interface.

**Full physical simulation system.** Full physical simulation system install the simulation software in a real PC to simulate the SDN major nodes. Its advantage is that SDN network can be completed without any platform. But the disadvantage is that it lack unified management and state inspection.

**Para-virtual simulation system proposed in the paper.** Based on the above analysis, the two simulation systems have certain shortcomings, so in this paper we proposed a new simulation system-para virtual simulation system. It can solve the performance issue caused by Lightweight virtual simulation system, while get over the management problem caused by Full physical simulation system.

### Build para-virtual simulation system

Para-virtual simulation system is divided into four levels. The underlying hardware are real physical devices. Virtual platform build a virtual node resource pool. Underlying software use Emulab to achieve the management and testing functions. Simulation software includes switch emulation software and controller emulation software which support OpenFlow protocol.

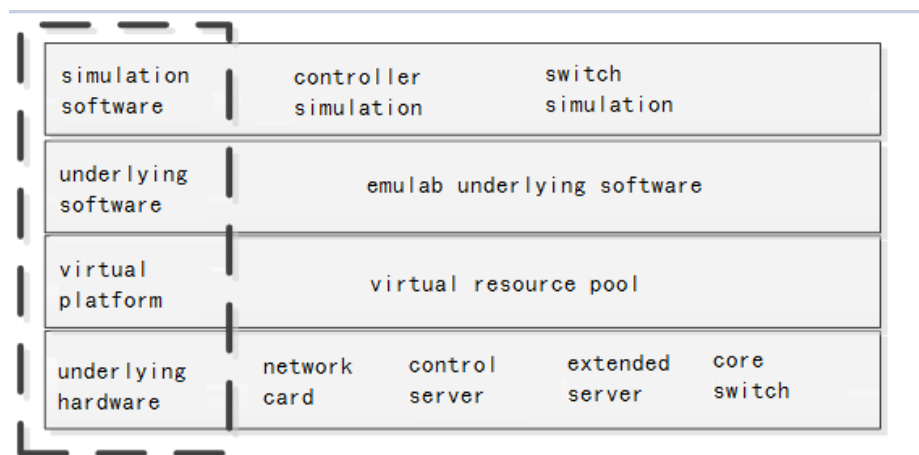


Fig. 1 para-virtual simulation system architecture

**The underlying hardware.** The underlying hardware of the system is divided into three parts. Control server is the core of the system which acts as manager. Core switch connect each server, it divided into two VLAN, VLAN10 acts as management area, VLAN1 acts as experimental area. Extended server mainly used to extend the experiment nodes.

**Virtual resource pool.** Virtual resource pool[5] contains several virtual nodes. Virtual node includes two types, one is BOSS, it connect network, ensure users accessible to the network. The other is OPS server which connect network and communicate with BOSS.

**Underlying software.** Emulab contains a series of management and testing tools, it provides a unified user interface for flexible experimental management and status display. Unified management and control module is the core of the software which allows to manage and control the lower simulation part. Display and interaction module is to interact with the user. The module is aware of the operational status of the entire system.

**3.4 Simulation Software**  
 (1) Controller simulation. The often used central controller emulation open-source software vary from each other. The system uses FloodLight[6] as the open-source controller software. As can be seen, FloodLight supports multi-thread and Web UI. More importantly, FloodLight has a commercial version which is good for the transplant of the system when expand.

(2) Switch simulation. The platform uses a pure software vSwitch, it can reduce costs and can simulate high-performance switches through virtual software. Moreover, it has the feature of flexible configuration and easily expansion.

### Build para-virtual simulation system.

(1) Start process

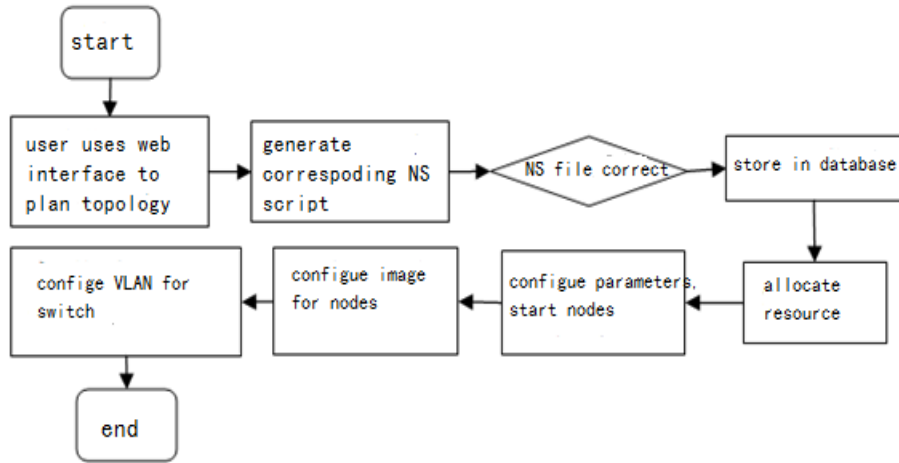


Fig. 2 para-virtual simulation system start process

## (2) End process

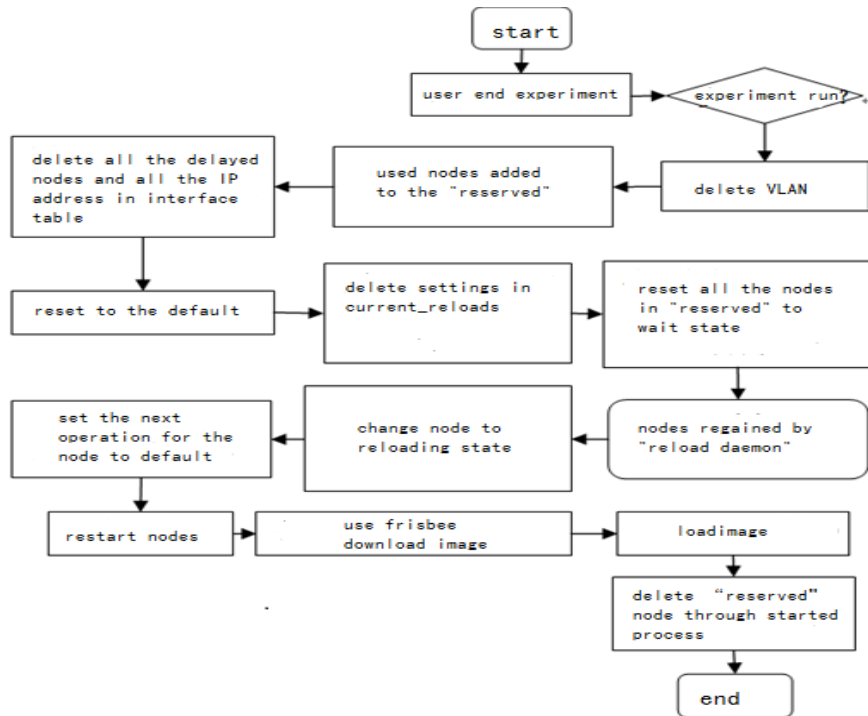


Fig. 3 para-virtual simulation system end process

## Verification test on para-virtual simulation system

We select a typical data forwarding scene to test para-virtual simulation system. The experiment requires that all packets from the switch 2th port to be forwarded to 3th port, and at the same time all packets from 3th port to be forwarded to 2th port.

(1) Resource Required. The resource required for this experiment shown in this table, it can be seen that the basic operating system is Ubuntu system. We install Wireshark to capture communication packet, analysis whether the simulation system works.

(2) Topology building. This experiment use the Ping command to send ICMP packet to test the network connectivity. First build network topology through the graphical interface in the simulation platform.

(3) Function verification. (a) forward packets. Test node connectivity via Ping command. We can see that 2th and 3th port devices can receive messages sent by each other.

```

root@host1:~# ping 10.1.1.2
PING 10.1.1.2 (10.1.1.2) 56(84) bytes of data.
64 bytes from 10.1.1.2: icmp_seq=1 ttl=64 time=3.61 ms
64 bytes from 10.1.1.2: icmp_seq=2 ttl=64 time=3.27 ms
^C
--- 10.1.1.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 3.278/3.444/3.610/0.166 ms
root@host1:~#

root@host2:~# ping 10.1.4.2
PING 10.1.4.2 (10.1.4.2) 56(84) bytes of data.
64 bytes from 10.1.4.2: icmp_seq=1 ttl=64 time=13.6 ms
64 bytes from 10.1.4.2: icmp_seq=2 ttl=64 time=3.18 ms
64 bytes from 10.1.4.2: icmp_seq=3 ttl=64 time=3.57 ms
^C
--- 10.1.4.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 3.181/6.805/13.663/4.851 ms
root@host2:~# ping 10.1.4.2

```

Fig. 4 Forwarding the data packet authentication

(b)drop packets. Test node connectivity via Ping command.We can see that 2th and 3th port devices can't receive messages sent by each other when we configure the drop rule.

```

root@vswitch:/home/wire# curl -d '{"switch": "00:00:f2:60:86:d6:74:4a", "name": "my-flow2", "cookie": "0", "priority": "2232", "ingress-port": "2", "active": "true", "actions": ""}' http://10.0.1.13:8080/wm/staticflowentrypusher/json
root@vswitch:/home/wire#

root@host1:~# ping 10.1.1.2
PING 10.1.1.2 (10.1.1.2) 56(84) bytes of data.
From 10.1.4.2 icmp_seq=1 Destination Host Unreachable
From 10.1.4.2 icmp_seq=2 Destination Host Unreachable
From 10.1.4.2 icmp_seq=3 Destination Host Unreachable
^C
--- 10.1.1.2 ping statistics ---
4 packets transmitted, 0 received, +3 errors, 100% packet loss, time 3015ms
, pipe 3
root@host1:~#

root@host2:~# ping 10.1.4.2
PING 10.1.4.2 (10.1.4.2) 56(84) bytes of data.
From 10.1.1.2 icmp_seq=1 Destination Host Unreachable
From 10.1.1.2 icmp_seq=2 Destination Host Unreachable
From 10.1.1.2 icmp_seq=3 Destination Host Unreachable
^C
--- 10.1.4.2 ping statistics ---
4 packets transmitted, 0 received, +3 errors, 100% packet loss, time 3016ms
, pipe 3
root@host2:~#

```

Fig. 5 Drop the packet authentication

## Summary

This article proposed an improved SDN simulation system use the para-virtual technology which overcomes the problems of the existing simulation systems.It has stronger interactive,higher performance and easier manageable features which is really promising.It can solve the performance issue caused by Lightweight virtual simulation system, while get over the management problem caused by Full physical simulation system. Then we build the para-virtual simulation system,finally we carry out the verification test on the simulation system.

## Acknowledgment

This work is supported by National Natural Science Foundation of ChinaProject(61302087, 61401038).

## References

- [1] Halpern. J. M, Standards collisions around SDN, IEEE Communications Magazine, Vol.52, pp.10-15, 2014.

- [2] Muhammad H. Raza, A Comparison of Software Defined Network (SDN) Implementation Strategies, *Procedia Computer Science*, Vol.32, pp.1050-1055, 2014.
- [3] John.W, Splitarchitecture: SDN for the carrier domain, *IEEE Communications Magazine*, Vol.52, pp.146-152, 2014.
- [4] Marcos Antonio de Siqueira, Providing Optical Network as a Service with Policy-based Transport SDN, *Journal of Network and Systems Management*, Vol.23, pp.360-373, 2015
- [5] Tan.W, SDN-enabled converged networks, *IEEE Wireless Communications Magazine*, Vol.21, pp.79-85, 2014.
- [6] Tourrilhes. Jean, SDN and OpenFlow Evolution: A Standards Perspective, *Computer*, Vol.47, pp.22-29, 2014.